Lecture 14

Plan:

1) Global min cut

2) min T-odd cut.

3) Next time: matroids.

→ Mechthild Stoer

Stoer-Wagner alg for global mincut.

Setup: • Let $G=(V,E)$ undirected,
• $u: E \to \mathbb{R}_{\geq 0}$ nonneg. edge costs.

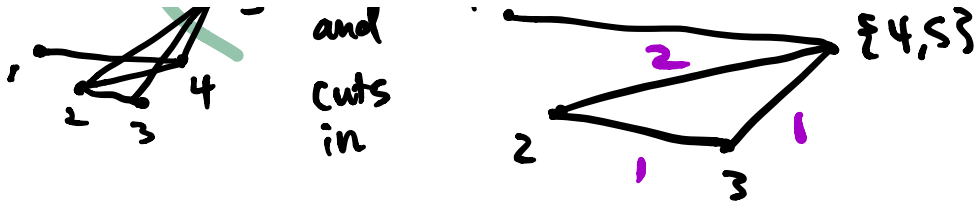
Algorithm idea: starting with arbitrary vertex,

- build "max adjacency ordering",
  i.e. greedily add the vertex w/ min cost to previous ones.

e.g. $u=1$

- Consider cut from last vertex, and also cuts obtained by shrinking last two vertices. (& recursing!)

e.g. $u=1$

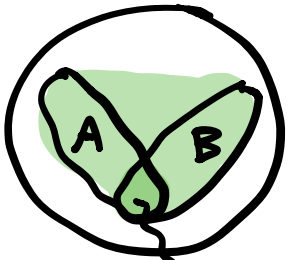


(duplicate edges get the sum of cost.).

- Claim: best cut found this way is global min cut.

Def: For $A, B \subseteq V$, define

$$w(A:B) := \sum_{\substack{i \in A \\ j \in B}} w((i,j)).$$



double.

Algorithm (Stoer-Wagner)

MINCUT(G) # outputs cut.

▷ Let v_1 any vertex of G

▷ $n := |V(G)|$

create ordering

- initialize $S =$
- for $i = 2 \dots n$:
  - $v_i = \underset{v \in V \setminus S}{\arg\min}\ u(S : \{v\})$
  - $S \leftarrow S \cup \{v_i\}$
- if $n = 2$:
  - return $\delta(\{v_n\})$.
- else:
  - Get G' by shrinking $\{v_{n-1}, v_n\}$

  # recursive call
  - Let $C = \text{MINCUT}(G')$
  - return less costly of C, $\delta(\{v_n\})$.

Analysis: uses a claim.

Claim: $\{v_n\}$ is a min v_{n-1}-v_n cut.

Claim ⇒ correctness:

- The min cut is either a min v_{n-1}-v_n cut, or not.
- If it is, claim ⇒ alg outputs it ✓
- If not, by induction on $n = |V(G)|$, algorithm outputs min cut in G'. ✓.

Proof of Claim:

Let $v_1, v_2, \ldots, v_{n-1}, v_n$ be the ordering from alg.

- $A_i :=$ sequence $v_1, \ldots, v_{i-1}$

- Consider candidate v_{n-1}-v_n cut, i.e. $C \subseteq V$ s.t. $v_{n-1} \in C$, $v_n \notin C$.

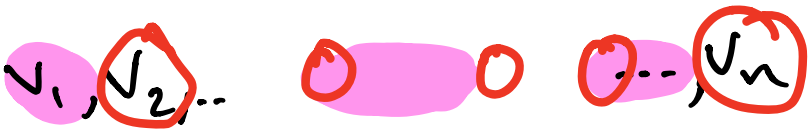
- Want to show

$$u(\delta(A_n)) \leq u(\delta(C)),$$

"$u(\delta(\{v_n\})$.

i.e. cut from $\{v_n\}$ is at least as good as C.

- define v_i to be **critical** if either v_i or v_{i-1} in C but not both.



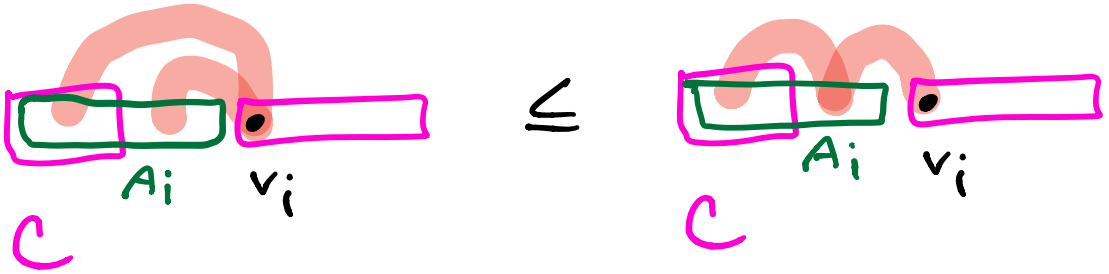
C critical

- **Subclaim:** Define $C_i := A_{i+1} \cap C$

If v_i critical, then

$$u(A_i : \{v_i\}) \leq u(C_i : A_{i+1} \setminus C_i) \quad \star$$

e.g.



replace $C_{i-1} : v_i$ by $C : A_i$.

Subclaim suffices:

because v_n is critical,

subclaim $\Rightarrow u(\delta(A_n)) \leq u(\delta(C))$

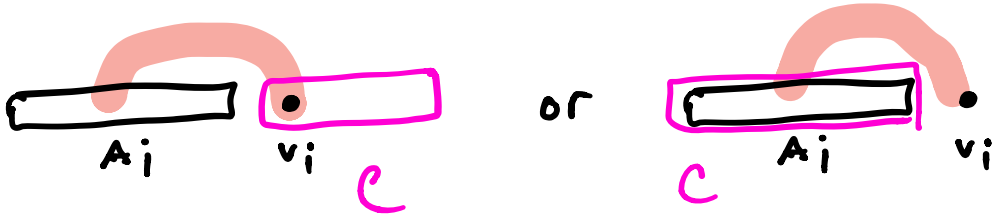
$u(A_n : v_n)$ $u(C_n : A_{n+1} \setminus C_n)$

is ✱ for $i = n$.

proof of subclaim:

by induction on seq. of critical vertices.

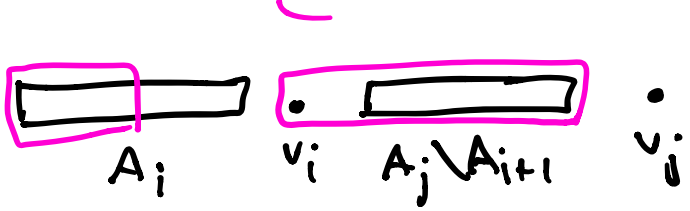
- (base:) ✱ true for first critical vertex v_i



✱ holds w/ inequality; nothing to replace.

- (inductive) Assume ✡ true for critical v_i, let v_j next critical.

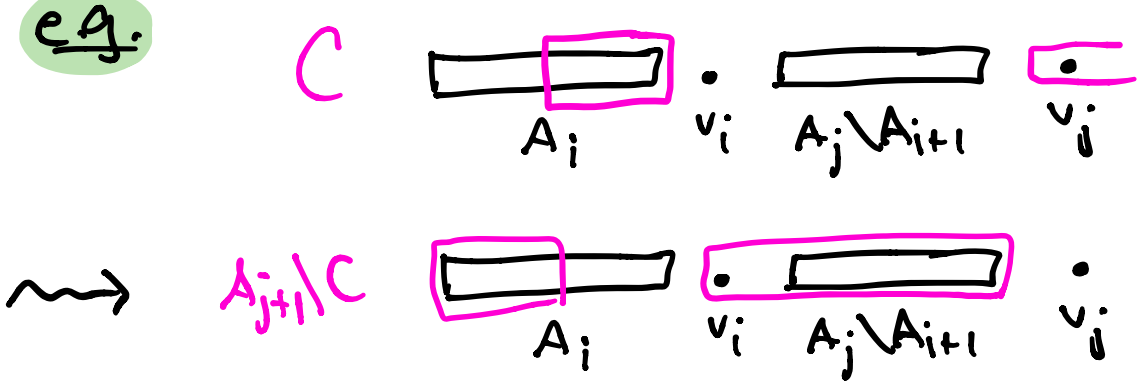
e.g.



- Assume $v_i \in C$, $v_j \notin C$. (as in pic)

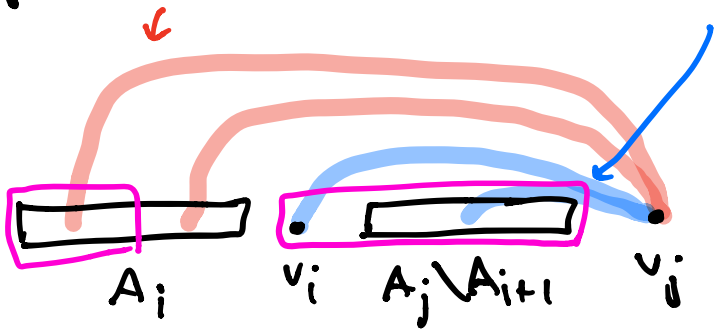
Is WLOG: if $v_i \notin C$, replace C by $A_{j+1} \setminus C$. RHS of ✡ unchanged.

e.g.



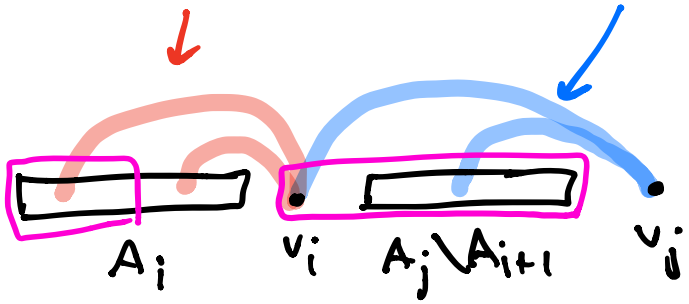
- Then

$$u(A_j : \{v_j\}) = u(A_i : \{v_j\}) + u(A_j \setminus A_i : \{v_j\})$$



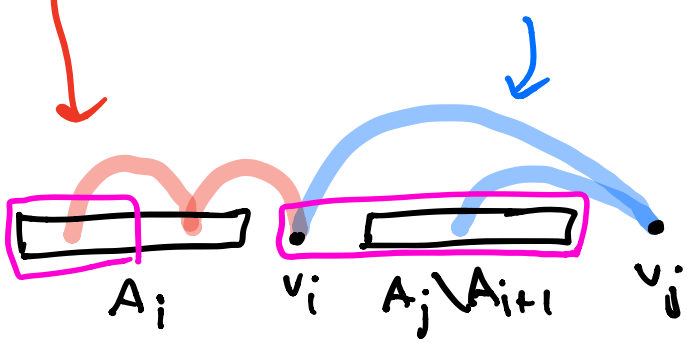
by ordering $\rightarrow$

$$\leq u(A_i : \{v_i\}) + u(A_j \setminus A_i : \{v_j\})$$



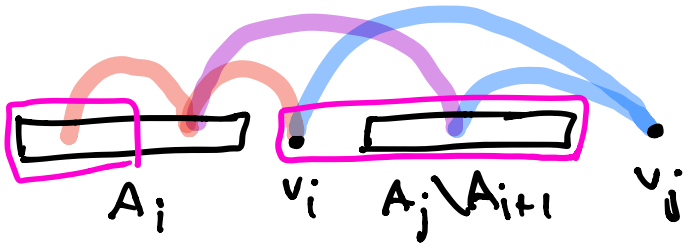
induction $\rightarrow$

$$\leq u(C_i : A_{i+1} \setminus C_i) + u(A_j \setminus A_i : \{v_j\})$$



$\leq u(C_j : A_{j+1} \setminus C_j)$.

j next critical.



purple contribution added; nothing removed b/c entire $A_j \setminus A_{i+1}$ in C. □

Running time:

Depends how you implement ordering.

- While building ordering, must maintain list of costs $c_{i+1}, \ldots, c_n$ $v_{i+1}, \ldots, v_n$ to $v_1 \ldots v_{i-1}$.

- must quickly find minimum & update new $c_{i+2}, c_{i+3}, \ldots, c_n$ after picking v_{i+1}.

- This is what "priority queues" are for, e.g. "Fibonacci heap".

- with Fibonacci heap, can build order in $O(m + \log n)$ time.

- total runtime
$$O(nm + n \log n).$$

- Compare to $\tilde{O}(mn^2)$ from computing $O(n)$ maxflows.

Submodularity

- Stoer-Wagner can be extended to minimize a more general class of functions than $S \mapsto w(\delta(S))$.

- function $f: 2^V \to \mathbb{R}$ **submodular** if $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$.

- **Examples:**
  - $f(S) = |S|$, "modular"
  - $f(S) =$ for w nonnegative, even if G directed.
  - V = food items on menu, $S \subseteq V$ meal
    $f(S)$ = "enjoyment of eating S"

Submodularity equivalent to "diminishing marginal returns":

For $S \subseteq T$, $v \notin S$,

$$f(S+v) - f(S) \leq f(T+v) - f(v)$$

- Stoer-Wagner algorithm can be extended to minimize any symmetric ($f(S) = f(V \setminus S)$) submodular function.

  ↳ Queyranne '95. (not covered here).

Application of submodularity:

Minimum T-odd cut

- $G=(V,E)$ undirected graph,
  $u: E \to \mathbb{R}$ nonnegative edge costs
  $T \subseteq V$ even size subset of V.

- **minimum T-odd cut problem:**

$$S = \arg\min_{S \subseteq V,\ |S \cap T| \text{ odd}} u(\delta(S)).$$

- Say S is **T-odd** if $|S \cap T|$ odd
- Note: S T-odd $\iff$ $V \setminus S$ T-odd
- why do we care? ~~Matching polytope~~!

Recall

THM (Edmonds) Let

$$X = \{1_M : M \text{ matching in } G\}.$$

Then $\text{conv}(X) = P$ where

$$P = \Big\{ x : \sum_{e \in \delta(v)} x_e \le 1 \quad \forall v \in V.$$

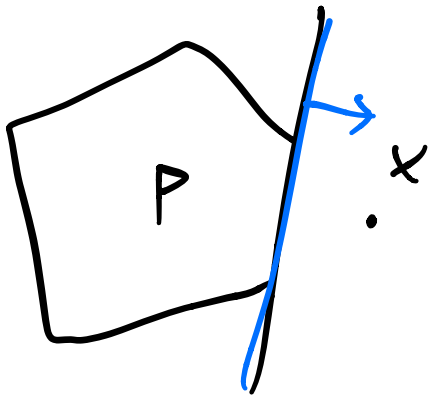
$$\sum_{e \in E(S)} x_e \le \frac{|S|-1}{2} \quad \forall S \subseteq V, \ |S| \text{ odd}$$

$$x_e \ge 0 \quad \forall e \in E. \Big\}$$

- How can we quickly test if $x \in P$? Exponential # constraints!?

- Padberg-Rao: Can express as min odd cut problem.

What's more, can get separating hyperplane if $x \notin P$.



- This can be used to optimize over P via ellipsoid alg, despite there being no polynomial size LP for optimizing over P (Rothvoss).

  We'll discuss more during ellipsoid.

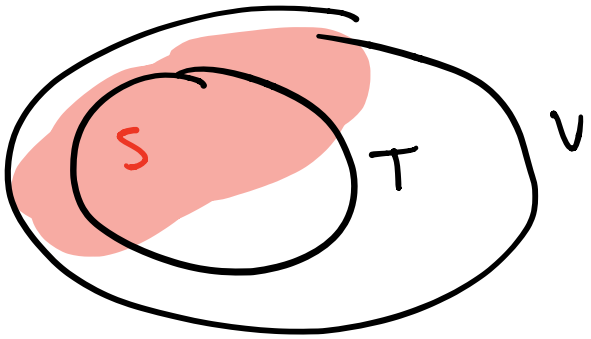
- **Today:** ▷ poly. time alg. for min T-odd cut
  ▷ crucially uses submodularity.

Algorithm ALG(G, T)

1) Find min cut among those with one vertex of T on each side:

$$S = \arg\min_{\emptyset \neq S \cap T \neq T} u(\delta(S)).$$ ☆

- Takes $|T|-1$ min s-t cut computations: fix $s \in T$, compute min s-t cut for all $t \in T \setminus s$.



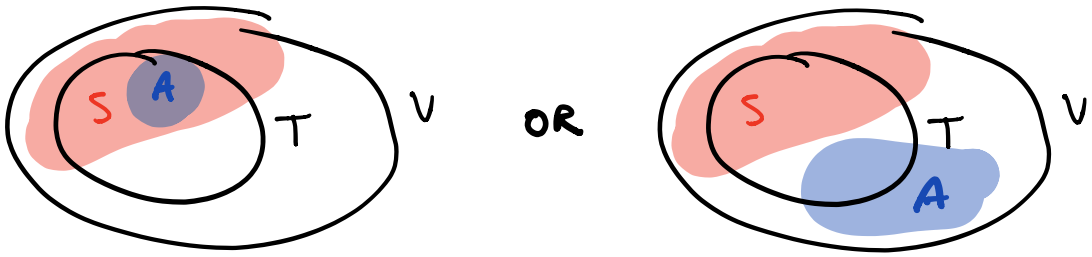
2) If S is T-odd cut, is minimum; return S.

Else: δf S T-even, use

Lemma: If S as in ✡, $|S \cap T|$ even,
$\exists$ min T-odd cut A with

$$A \subseteq S$$

or $$A \subseteq V \setminus S.$$



Pf: after alg.

- Lemma ⇒ 2 recursive calls suffice:

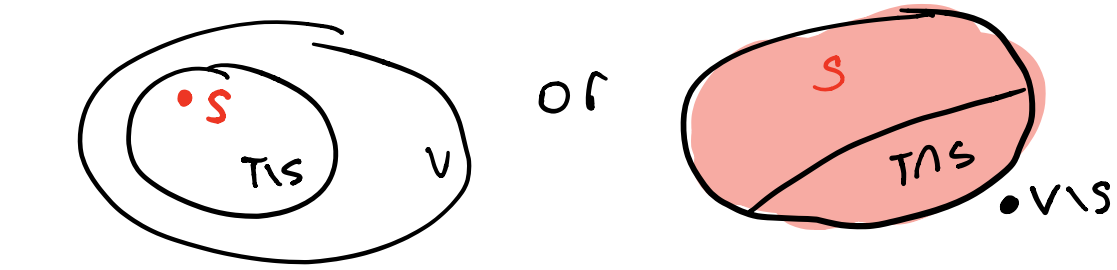
▷ $G_1 := G/S$ (shrink S to single vertex),

$T_1 := T \setminus S$.

▷ $G_2 := G/(V \setminus S)$

$T_2 := T \setminus (V \setminus S) = T \cap S$.

e.g.



Return:

$\min$ $ALG(G_1, T_1), ALG(G_2, T_2)$.

Running time why polytime if 2 recursive calls?

$R(k) :=$ largest possible running time with $|T| = k$, $|V| \leq n$

Then

a) $R(2) = T :=$ time for min s-t cut

b) $R(k) \le \max_{\substack{k_1 \ge 2 \\ k_2 \ge 2 \\ k_1 + k_2 = k}} \left((k-1)T + R(k_1) + R(k_2) \right)$.

sizes of $T \cap S$, $T \cap \bar{S}$ (must be even).

Step 1 (min s-t cuts).

Step 2 (recursive calls)

By induction, $R(k) \le k^2 T$.

Pf: • base: True for $k = 2$ ✓.

• inductive:

$$R(k) \le \max_{\substack{k_1 \ge 2 \\ k_2 \ge 2 \\ k_1 + k_2 = k}} \left((k-1)T + R(k_1) + R(k_2) \right)$$

$$\le (k-1)T + 4T + (k-2)^2 T$$

$$= (k^2 - 3k + 7)T$$

$\max k_1^2 + k_2^2$ s.t. $k_1 + k_2 = k$, $k_1 \ge 2$,

$k_2 \geq 2$

$= 4 + (k_2 - 2)^2 \cdot \leq k^2 T$

for $k \geq 4$. (k even).

Thus: algorithm is polynomial.

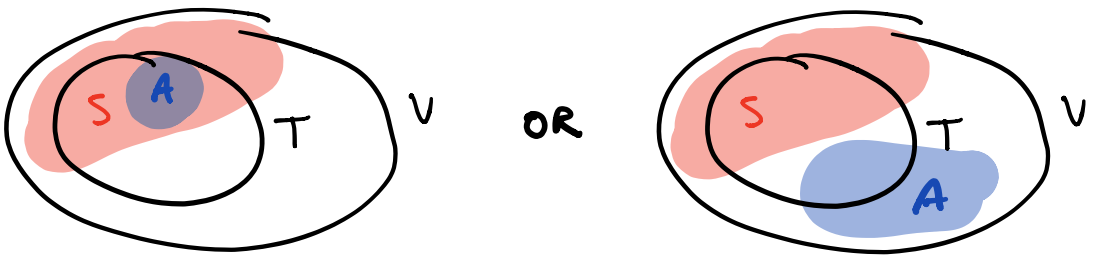
- Now for the lemma.
- Proof uses submodularity.

Lemma: Let S min cut subject to $\emptyset \neq S \cap T \neq T$, $|S \cap T|$ even, then

$\exists$ min T-odd cut A with

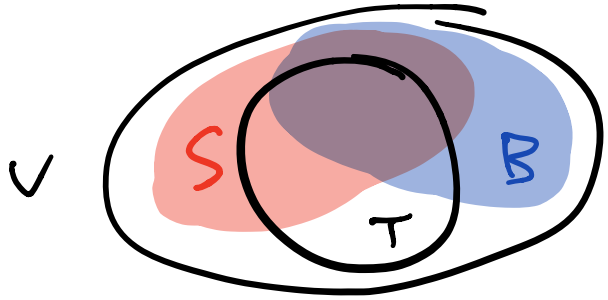
$A \subseteq S$

or $A \subseteq V \setminus S$.



Proof

- Let B be any minimum T-odd cut.

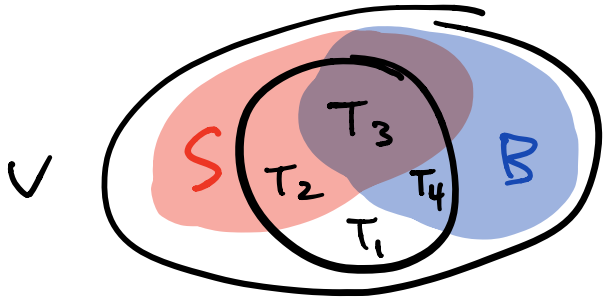


- We'll show we can take $A = S \cup B$ or $A = S \cap B$.

- Make a partition.

$$T_1 = T \setminus B \cup S \quad , \quad T_2 = (T \cap S) \setminus B$$

$$T_3 = T \cap B \cap S \quad , \quad T_4 = (T \cap B) \setminus S$$



- By definition of B, S, know all pairwise unions nonempty:

$$\left.\begin{aligned} T_1 \cup T_4 &= \emptyset \\ T_2 \cup T_3 &\neq \emptyset \end{aligned}\right\} \text{ by } \emptyset \neq S \cap T \neq T$$

$$\left.\begin{aligned} T_2 \cup T_1 &\neq \emptyset \\ T_3 \cup T_4 &\neq \emptyset \end{aligned}\right\} \text{ by } |B \cap T| \text{ odd, } |T| \text{ even.}$$

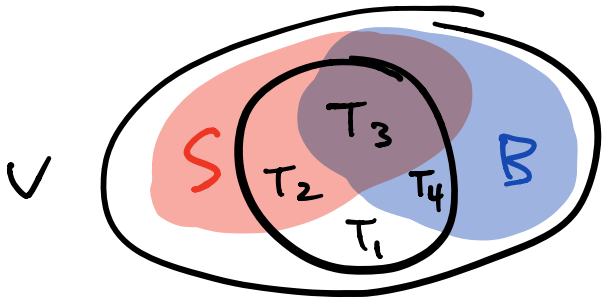
$\Rightarrow$ either T_1, T_3 nonempty or T_2, T_4 nonempty.

- By possibly replacing $B \leftarrow V \setminus B$, may assume T_2, T_3 nonempty.

- submodularity of cut $\Rightarrow$

$$(\Delta) \quad u(\delta(S)) + u(\delta(B)) \geq u(\delta(S \cup B)) + u(\delta(S \cap B)).$$

- As $T_1 \neq \emptyset$, $T_3 \neq \emptyset$, $S \cap B$ & $S \cap B$ separate vertices of T.



- One of $S \cup B$, $S \cap B$ is T-even & the other T-odd because

$$|(S \cap B) \cap T| + |(S \cup B) \cap T| = |T_2| + 2|T_3| + |T_4|$$

$$= |S \cap T| + |B \cap T| \text{ is odd.}$$

Summary: one of $S \cup B$ is candidate "S", the other is candidate "B".

- $\triangle \Rightarrow$ whichever of $S \cup B$, $S \cap B$ odd has cut value $\leq u(\delta(B))$, the other $\leq u(\delta(S))$.

(else the other violates minimality of B or S due to $\triangle$)

$\Rightarrow$ Either $S \cap B$, $S \cup B$ is min T-odd cut. $\square$